Computational Geometry

- Project Report

李国志 994912
赵建伟 994913
王 颀 994914

1、 INTRODUCTION

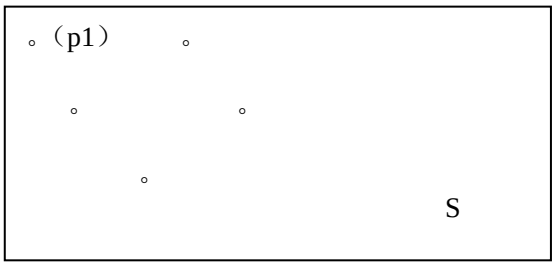
设 S 为平面上给定的点集 $S=\{P_1, P_2, ...P_N\}$,则判断 S 能否被以 r 为半径的圆所覆盖的关键就是以 P_i(i=1..N)为圆心, 以 r 为半径的所有的圆的交集不为空。即对于 $B_i=\{Pi,r\}$, 则 $\bigcap_{i=1}^{n} B_i \neq \Phi$. 2_center 问题则是寻找两个完全相同的圆将 S 完全覆盖,并且要求着这两个圆的半径要尽可能的小. 2_center 问题是 p_ center 问题中较有实际意义的一个,因此,近几年来已有多篇文章描述了该问题的研究状况,并且当前该问题的算法复杂度已达到了 $O(n\log^2 n)$。

2、 Previous Works

在以前的 2_center 问题的解决方案中, 基本的沿用了前面描述的判断给定的 S 能否被某一圆所覆盖的算法思想。1997 年, Telaviv 大学的 M.Sharir 则利用一个 2 叉平衡树的数据结构来加速上述问题的判定,并且在他的算法中由于点集 S 的动态维护以及 Parametric Searching 机制的应用,算法复杂度达到了 $O(n\log^9 n)$。随后 D.Eppstein 对 M.Sharir 的算法进行了改进,并使算法复杂度达到了 $O(n\log^2 n)$。

3、 Our Works

对于平面上给定的点集 S, 如图



如果要找到两个完全相同的圆将 S 完全覆盖, 我们的思路如下:

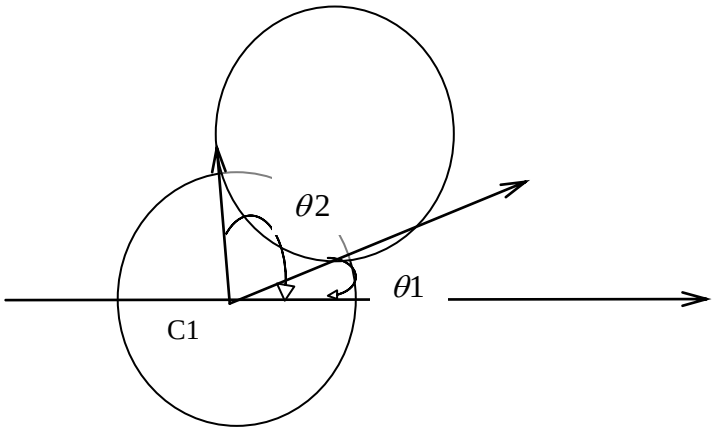
3.1 确定 r 的大小

1).将给定点集 S 内的所有点 P_i 按 X 轴排序,然后取 S 内最左边点与其最远点的距离为半径 r ,在这种情况下,用一个半径为 r 的

圆显然可将S完全覆盖.

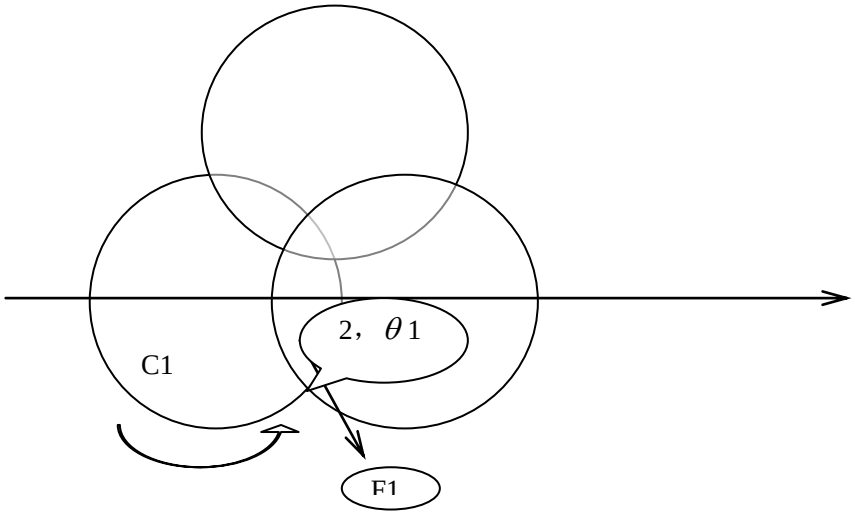
2).用二分法,初始化MIN=0,MAX= r, r=(MAX+MIN)/2.以S内最左边点p1为圆心,以r为半径作圆c1,判断c1能覆盖哪些点,然后,基于同样的算法,判断剩下的点是否能被另一半径为r的圆所覆盖,如不能,则取MIN= r, r=(MIN+MAX)/2,重复上述判断;否则,重复2),直到r最小为止.

3.2 数据结构的表示



如图,以r为半径,分别以p1, …, pn为圆心作圆。设C1 = C(p1, r)与其它圆相交于m个点。将这些点按θ排序, 则ci与c1交点可表示为（I, θ1),（I, θ2), 其中,（ $-\pi \le \theta 1 \le \theta 2 \le \pi$)。

3.3 对c的位置进行讨论



设c1与(n-1)个圆相交于m个交点I={I1, I2, …Im}(如相切则算两个交点),则c1
被分为个face ,F={f1, f2, …fm}。因为要寻找的两个圆的圆心c, c'

至少有一点位于 c1 内,不妨设为 c,由于在 c1 与各圆相交构成的任一 face 内(如 f1 内)内的任一点所作的圆所覆盖的点数相等，即如果在 f1 中的 任一点所作的圆所覆盖的点数与在（2，θ1)所作的圆覆盖的点数相等。这样如果第一个圆的圆心 c 在 c1 内，则必可在 C1 上找到一点所作的圆与在 c 点所作的圆覆盖的点数相等，也就是在 C1 上定可找到一点作为第一个圆的圆心。

3．4．c 位置的确定

将$\theta=-\pi$作为起始位置，并用一个 n 元数组 flag[n]作为图中各点的标志位（初始值均设为 0)，如图中箭头所示的方向沿 C1 搜索，当遇到 C1 与其他圆的某一个交点时，取出该点所属的 I 值，并将 flag[I]置反。如果该点的 flag[I]置反前为 1，说明以 Ii 为圆心，以 r 为半径的圆覆盖的点数开始减少（排除了 Pi)，此时统计 flag 数组中为 0 元素的位置 i，并判断所有 Pi 是否能被另一个半径为 r 的言所覆盖。如果能，则重复步骤 3.1(2),取新的 r 值,返回 3.4;否则,从 Ii 点沿 C1 继续搜索,重复 3.4 直到 C1 上的所有交点都被遍历,即可确定 c 的位置.

3.5.　根据 flag 中为 0 元素的位置 i,判定所有 Pi 是否能被另一以 r 为半径的圆所覆盖

仍如上法,将剩下的点按 X 轴排序,重复 3.4,最后判断 flag 数组中内容为 1 元素的个数,如等于剩下的点数,则说明剩下的点能被另一半径为 r 的圆所覆盖,圆心 c'也可由上法确定,否则,点集 S 中的所有点不能被半径为 r 的两个圆覆盖.

3.6 重复 3.1(2)和 3.4,直到 r 取得最小值为止.

4 算法复杂度分析.

由于 c1 与 ci 相交于 m 个交点(m <= 2*(n-1)),遍历每个交点(O(n))，在最坏情况下，每次都需判定未被第一个圆覆盖的点（最多 n-1 个）能否被另一圆所覆盖，时间复杂度为 O(nlogn)。所以，在最坏情况下,该算法的时间复杂度 为 O(n)*O(nlogn) = O(n^2logn)。

5 Reference:

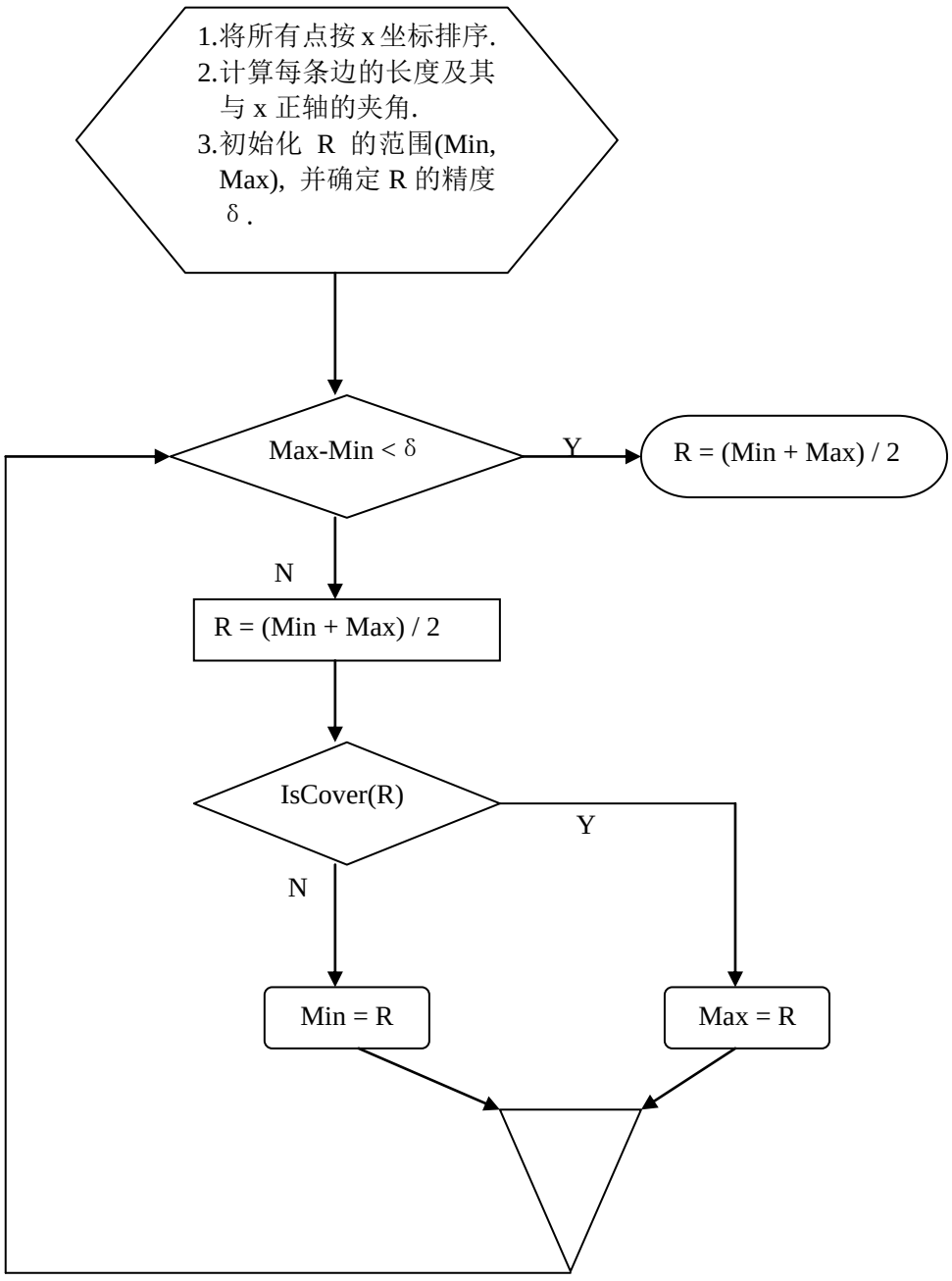
1.A Near-Linear Algorithm for the Planar 2-Center Problem M.Sharir Discrete Compute Geom 18:125-134(1997)

2. The Discrete 2-Center Problem /P.K.Agarwal, M.Shariar,and E.Welzl Discrete Comput Geom 20:287-305(1998)

3. Finding Tailored Partitions John Hershberger and Subhash Suri Journal Of Algorithm 12 ,431-463(1994)

附：流程图

Main


1.将所有点按 x 坐标排序.
2.计算每条边的长度及其与 x 正轴的夹角.
3.初始化 R 的范围(Min, Max), 并确定 R 的精度 δ.
Max-Min < δ
Y
R = (Min + Max) / 2
N
R = (Min + Max) / 2
IsCover(R)
Y
N
Min = R
Max = R

IsCover

/* To decide whether there are two disks with R as radius that cover the given points */